Department of Mathematical Sciences and Informatics. UIB.
Technical report A-03-2007

Enhancing the response of ReCANcentrate in presence of faults

Manuel Barranco, Julián Proenza
Dpt. Ciències Matemàtiques i Informàtica
Universitat de les Illes Balears, Spain
manuel.barranco@uib.es, julian.proenza@uib.es

Luís Almeida
DET/IEETA
Universidade de Aveiro, Portugal
lda@det.ua.pt

September 13, 2007

Abstract

Safety-critical applications require high dependable communication infrastructures. In this context, the use of CAN has been controversial due to dependability limitations. To overcome some of those limitations, we have proposed a replicated star topology, ReCANcentrate, whose hubs incorporate the necessary fault-treatment and fault tolerance mechanisms. Additionally, ReCANcentrate strongly simplifies the management of the replicated media that each node performs. In this document we propose additional mechanisms to enhance the fault tolerance of ReCANcentrate and to enable graceful degradation in presence of faults.

1 Introduction

Distributed embedded control systems for safety-critical applications, e.g. X-by-Wire systems, require high-dependable communication infrastructures [1]. In this context, there has been a growing interest in using CAN [2], due to its advantages related to dependability and real-time response. Nevertheless, the use of CAN in critical applications has been controversial due to dependability limitations. Some of them arise from its non-redundant bus topology, which lacks the necessary error-containment and fault tolerance mechanisms. To overcome these limitations, we have developed a new replicated star topology, called ReCANcentrate that includes two hubs [3]. As depicted in Figure 1, each node is connected to each hub by a dedicated link that contains an uplink and a downlink. Both hubs are interconnected by means of at least two *interlinks* each of which contains two independent sublinks, one for each direction. The hubs exchange their traffic through the interlinks and they couple to each other [3], so that both hubs behave as one, i.e. a hub coupling is enforced. Additionally, each

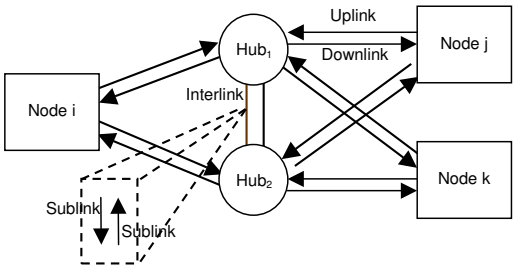

Figure 1: Architecture of ReCANcentrate

hub includes fault-treatment capabilities to contain errors originated at nodes, and to provide tolerance to hub, link and interlink faults. ReCANcentrate is fully compatible with CAN and commercial off-the-shelf (COTS) CAN components, being transparent for any CAN-based application.

Notice that ReCANcentrate uses active replicated media in order to provide fault tolerance. For this purpose, the same data is transmitted in parallel through each of the media replicas; so that, in principle, each communication medium, i.e. each star, can be considered as a channel that conveys a replica of the same data. Two main problems arise when managing active replicated channels in parallel. First, each node must be able to deal with redundant frames; basically, it needs to detected omissions and duplicates [4]. Second, each node must be able to detect when a fault in the media prevents it from communicating through a given medium; so that the node can continue communicating using only non-faulty medium replicas. Fortunately, the hub coupling creates a single logical broadcast domain, so that nodes can use a simple replicated media management [4].

Although the hub coupling is fault tolerant due to the use of more than one interlink, in this document we propose mechanisms to enhance the response of ReCANcentrate in presence of faults. First, these mechanisms allow to consistently recover from transient hub decouplings. Second, they enforce that, when the hub coupling is definitively lost, both hubs and all nodes can consistently decide to communicate using two independent broadcast domains, thereby enabling graceful degradation.

2 ReCANcentrate basics

The physical layer of CAN implements a wired-AND function of every node contribution, thereby providing a dominant/recessive transmission [2]. Additionally, CAN communication relies on a complex bit synchronization mechanism that guarantees that nodes have a quasi-simultaneous view of every single bit on the medium. This allows the definition of mechanisms [2] that improve CAN's dependability and real-time response.

However, one of the main impediments for using CAN in safety-critical control systems is that it relies on a bus topology with scarce error-containment and fault tolerance mechanisms. For this reason, a CAN bus includes multiple single points of failure,

i.e. components whose failure cause the failure of the overall system [3]. Particularly, the faults that may cause a generalized failure in CAN are [3]: stuck-at-dominant, stuck-at-recessive and bit-flipping, which can occur within nodes or in the medium; medium partition; and babbling idiot.

In order to eliminate all single points of failure from a CAN network we have developed a new replicated star topology called ReCANcentrate [3] (see Figure 1). The replication strategy is such that nodes transmit the same data through both stars in parallel. Internally, each hub performs an AND coupling in two stages. In a first stage each hub receives each node's contribution through the corresponding uplink and couples them, thereby obtaining what is called the contribution of that hub. This contribution is sent to the other hub using one sublink of each interlink. In a second stage each hub couples the contribution received from the other hub with its own contribution, and broadcasts, through the downlinks, the resulting signal to the nodes that are attached to it. These couplings create a single logical broadcast domain, since all hubs behave like one, transmitting the same value bit by bit through their downlinks, i.e. nodes have a quasi-simultaneous view of every single bit on the communication domain [3].

Errors generated by a fault can corrupt the frame being broadcast and lead all nodes to lose the bit synchronization. ReCANcentrate uses the same mechanism as standard CAN to enforce data consistency in such cases. When a node or a hub detects an erroneous bit, it globalizes the error by signalling an error-frame, which compels all nodes to reject the frame that is being broadcast and to signal the error too. During the error-signalling all nodes become synchronized again, so that the rejected frame can be transmitted again once the signalling is finished. Nevertheless, if a permanent fault is not correctly isolated, nodes will be signaling errors until they become disconnected. Fortunately, the use of an uplink and a downlink within each link and the use of two independent sublinks in each interlink, allow each hub to monitor the contribution of each node and of the other hub separately and diagnose the localization of any fault. Permanent stuck-at or bit-flipping contributions are disabled, and so not coupled, thus being confined to the port of origin.

Finally, for the sake of survivability, each hub uses a specific restoration policy to re-couple any port contribution, after a given time period during which no errors are received through that port.

2.1 Replicated media management at each node

Since hubs are coupled and enforce a single communication domain, each node does not have to deal with a set of replicated channels, but with different views of the same channel, which is easier. Specifically, each node can manage the replicated media by simply performing each transmission through one of the media replicas, i.e. through one hub, only, while receiving from both hubs at the same time. The node is constituted by COTS components only: two CAN controllers and a micro-controller (Figure 2). Each CAN controller is connected only to one hub by means of a dedicated uplink and downlink, using for this purpose two COTS transceivers [3]. Each node can easily remove duplicates and detect omissions, since it is expected that both controllers quasi-simultaneously notify of each frame exchanged on the network [4]

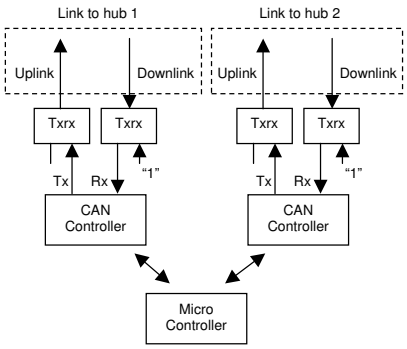

Figure 2: Node hardware architecture

Additionally, it is simplified how each node treats any fault occurring in the media, i.e. hub, connectors, cables, etc. A fault in the media can only lead one or more nodes to observe that one of its two controllers omits notifications of transmissions and of receptions, or accumulates too many errors. In the first case, the fault is tolerated since the node continues communicating through the other controller. In the second case, the node uses the typical CAN fault diagnosis mechanisms to diagnose that the impaired controller cannot communicate. Specifically, this happens whenever the *Transmission Error Counter* (TEC) [2] or the *Reception Error Counter* (REC) reaches a programmable threshold, called *error warning limit*.

3 Fault response enhancements

The referred simple media management is possible under the hypothesis that there is a single communication domain, i.e. as long as hubs are coupled. In order to enforce this hypothesis even in presence of faults, hubs are interconnected by means of more than one interlink, thereby tolerating sublink and sublink ports failures. Moreover, each hub has the ability to restore transiently faulty sublinks, which increases interlink survivability. In this document we propose to use this ability of restoring sublinks to recover from situations in which both hubs are decoupled, but there are, at least, two sublinks (that would allow hubs to exchange their traffic) whose failure is only transient. Additionally, we want to enable graceful degradation when hubs become permanently decoupled, since nodes could still use a different media management strategy, e.g. they could use both stars independently. In order to achieve these two objectives, it is necessary to further enhance the response of ReCANcentrate in front of faults.

The first enhancement is to prevent nodes from communicating as long as a hub decoupling is not detected and treated. Otherwise, nodes will manage the replicated media incorrectly believing that there is a single communication domain, which might lead to undesired situations, e.g. an inconsistency will occur when a frame sent by a node is not received by a subset of nodes that might have lost the connection with the hub the frame is sent to.

Taking into account the ReCANcentrate's fault model, hubs become decoupled in three cases. First, when at least one hub isolates all its incoming sublinks, due to stuck-at or bit-flipping faults. A node has no mechanism to detect such a situation, and although the hub that isolates all sublinks knows that a decoupling took place, currently it has no mechanisms to notify nodes of it. Moreover, even if hubs had this kind of mechanism, nothing ensures that the other hub also performs a decoupling (or that both hubs decouple each other at the same time), so that nodes could be inconsistently informed about a decoupling. The second case in which hubs become decoupled is when all the incoming sublinks that a hub was considering as non faulty suffer a stuck-at-recessive fault. In contrast to what happens with a stuck-at-dominant or a bit-flipping, a stuck-at-recessive fault does not necessarily generate errors that block the communications. Thus, nodes might incorrectly believe that there is a single communication domain as long as hubs do not diagnose the stuck-at-recessive. Finally, we want to introduce a third case of hub decoupling by adding some malicious faults to the hub's fault model. We consider that a fault in the internal circuitry of the hub may compel it not to couple the contribution that comes from the other hub or/and not to send its own contribution to that hub. Furthermore, it is even possible that a hub fails by not detecting a hub decoupling when it has isolated all its incoming sublinks.

The second necessary enhancement is to enforce that both hubs and all nodes agree on whether it is possible to recover from a hub decoupling, or only graceful degradation is viable. Notice that we focus on how to enforce such agreement; the specific strategy nodes use to communicate using two decoupled hubs is beyond of scope.

4 ACK interdependence

We have devised a new mechanism that blocks the communication in both stars when hubs become decoupled, in order to prevent any node from communicating before the decoupling is detected and treated. This mechanism relies on the fact that, in CAN, each frame must be acknowledged by at least one receiver node sending a dominant bit during the *ACK slot* [2]. Otherwise, the transmitter observes a recessive bit during the ACK slot, aborts the transmission of the frame signaling an error, and tries to retransmit it. We have modified the behavior of the hub, so that during the ACK slot of every frame it only broadcasts a dominant bit to its own nodes, if that dominant bit is actually received in the contribution from the other hub. In other words, the hub does not propagate the ACK bit received from its nodes, but only the ACK bit received from the nodes connected to the other hub. Thereby, no frame will be exchanged in any star if both stars are not correctly coupled and synchronized, i.e. an *ACK interdependence* is created between both stars.

Additionally, whenever hubs become definitively decoupled, each one of them changes its mode of operating, to broadcast the ACK bits received from its own nodes. This allows nodes to use both stars independently.

Notice that the *ACK interdependence* mechanism fulfills the first enhancement specified in Section 3; even in presence of the new malicious hub faults we have added to our fault model. This is because the *ACK interdependence* makes it not necessary that any hub detects a decoupling to prevent nodes from communicating.

Nevertheless, the *ACK interdependence* mechanism relies on a new hypothesis: no hub does propagate the ACK bits of its own nodes. Thus, it is necessary to restrict the failure semantics of the hub to ensure that it cannot maliciously fail by not fulfilling this hypothesis. For this purpose, we are devising a kind of bus-guardian for each hub that we call *ACK Unit*. This unit will prevent its corresponding hub from incorrectly broadcasting the ACK bits received from its own nodes. An important requirement has been imposed to the design of the *ACK Unit*: to guarantee the failure independence between this unit and its corresponding hub.

5 Consistent Network Reconfiguration

The second enhancement specified in Section 3 claims for a mechanism that ensures that, when a hub decoupling occurs, nodes and the hubs agree on whether the single communication domain can be reestablished, or only graceful degradation is viable because hubs are definitively decoupled. To achieve this, we have proposed a new protocol called *Network Reconfiguration Agreement Protocol*. The basic idea underlying it is that each hub must be able to notify of its presence by means of a frame whose identifier is reserved to that hub, i.e. by means of an *Identification Frame*. Moreover, in order to ensure that a hub can notify of its presence when needed, the two *Identification Frames* respectively have the two highest priority frame identifiers.

Figure 3 depicts an sketch of the *Network Reconfiguration Agreement Protocol*. The first phase is the *Sublink Restoration Phase*, during which both hubs consistently decide whether or not they couple again. It can be summarized as follows. Firstly, the hub compels the other hub to also execute this phase, by forcing it to isolate all its incoming sublinks. Then it forces all CAN controllers connected to it to enter into the *bus-off state*; so that they will not try to communicate. Secondly, hubs test the sublinks in order to decide whether or not it is possible to communicate with each other. For this, each hub restores its incoming sublinks that are silent, i.e. sublinks from which it only receives recessive bits, and tries to alternatively exchange with the other hub its *Identification frame*. If after a given interval of time, hubs are coupled again, they execute the *Node Restoration Phase*. Otherwise, they execute the *Definitive Decoupling Signalling Phase*.

The *Node Restoration Phase* aims at joining nodes in such a way that, when it finishes, all joined nodes can consistently start to communicate at the same time using the single communication domain. First, the hub restores the ports corresponding to all CAN controllers that have been silent during the *Sublink Restoration Phase*. Then, hubs cooperate to constantly broadcast through the whole communication domain their *Identification frames* alternatively. If after a given interval of time, hubs have monitored a given number of *Identification frames* pairs, the phase successfully finishes. At this point, each hub has only kept restored controllers that have not generated errors and that have acknowledged any *Identification frame* of the other hub, i.e. only nodes that have detected the reestablishment of the single communication domain.

Notice that since data consistency applies to the communication between both hubs and, when hubs are coupled, to all the communication domain, hubs consistently reach the end of these two phases at the same time. Moreover, during the *Node Restora-*

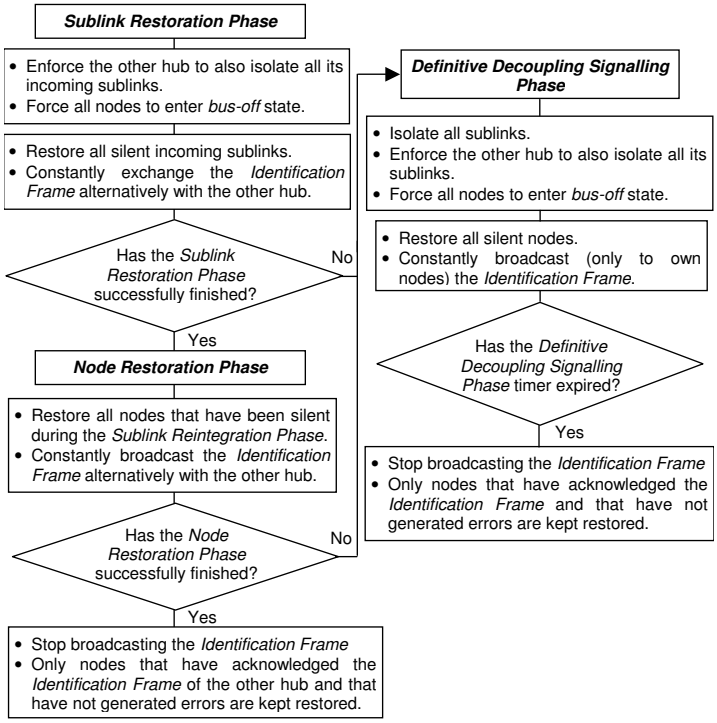

Figure 3: Network Reconf. Agreement

tion Phase, nodes cannot communicate with each other, since the *Identification frames* block their access to the media. This enforces that restored nodes can consistently start to communicate at the same time when this phase finishes.

The *Definitive Decoupling Signalling Phase* is devoted to enforcing that all nodes agree on that it is not possible to reestablish the single communication domain. In a first step, each hub forces the other hub to also execute this phase, and all CAN controllers connected to it to enter into the bus-off state. In a second stage, the hub restores the ports corresponding to the controllers that have been silent during the first step, and starts to constantly send its own *Identification frame* to them. A hub finishes the phase whenever it has successfully sent a given number of *Identification frames*, or when the timer that limits the duration of this phase expires. At this point, the hub has only kept restored the ports corresponding to controllers that have not generated errors, and that have acknowledged the transmissions of its *Identification frame*. Notice that during the second stage of this phase, the hub blocks the communication by constantly transmitting its *Identification frame*. This guarantees that all restored controllers can consistently star to communicate through the hub at the same time when the phase finishes. However, hubs are not coupled and, hence, this phase can evolve in a different way in both stars. Thus, it can be guaranteed neither that both stars become available

for communicating at the same time, nor that the same nodes can communicate through each of them.

Finally, in order to make it possible the *Network Reconfiguration Agreement Protocol*, we need to restrict the failure semantics of the hub, so that it cannot forge the *Identification Frame* of the other hub. For this purpose, the *Identification Frame* must be completely pre-builded, e.g. recorded in a ROM. Thereby, the probability that faults lead a hub to successfully transmit a frame other than its *Identification Frame* is negligible; because it could only happen if faults altere also the CRC field in a way that its value is correct with respect to the other frame fields. Additionally, each hub must be able of detecting and isolating any node that sends a frame with any of the identifiers reserved for the *Identification frames*.

6 Conclusions

In this document we propose two new mechanisms for ReCANcentrate that allow to consistently recover from a transient hub decoupling, and to enable graceful degradation when hubs become permanently decoupled. On the one hand, these mechanisms ensure data consistency at the frame level when a hub decoupling has not been already detected; while hubs try to reestablish the hub coupling; as well as when hubs and nodes decide that the coupling is definitively lost. On the other hand, one of these mechanisms enforces that both hubs and all nodes agree on whether it is possible to recover from a hub decoupling or only graceful degradation is viable. Furthermore, we have identified the failure semantics restrictions of hubs and nodes that are required by such mechanisms, and we have briefly pointed out how to fulfill these requirements.

References

[1] H. Kopetz and G. Grunsteidl, "TTP - A Protocol for Fault-Tolerant and Real-Time Systems," *IEEE COMPUTER*, January 1994.

[2] ISO, "ISO11898. Road vehicles - Interchange of digital information - Controller area network (CAN) for high-speed communication," 1993.

[3] M. Barranco, L.Almeida, and J. Proenza, "ReCANcentrate: A replicated star topology for CAN networks," *ETFA 2005. 10^{th} IEEE International Conference on Emerging Technologies and Factory Automation, Catania, Italy*, 2005.

[4] M. Barranco, J. Proenza, and L.Almeida, "Management of Media Replication in ReCANcentrate," *Submitted to: EUC 2007. The 2007 IFIP International Conference on Embedded and Ubiquitous Computing, Taipei, Taiwan*, 2007.